The Paninian Approach to Natural Language Processing

Subhash C. Kak
Department of Electrical and Computer Engineering, Louisiana State University

ABSTRACT

This article reviews the Paninian approach to natural language processing (NLP) and compares it with the current computer-based understanding systems. It is argued that Paninian-style generative rules and meta-rules could assist in further advances in NLP.

KEYWORDS: *natural language processing, knowledge representation, computer understanding systems, grammars, generative rules, the Paninian approach*

INTRODUCTION

Computer processing of a natural language, as opposed to an artificial language of the type used in writing computer programs, has gone through two distinct phases in the past 35 years. It was first thought that machine translation (MT) of one language into another should be an easy matter once one had compiled dictionaries and obtained mathematical representation of the grammars of the languages in question. It was believed that actual translation would proceed by replacing the words in the text by their equivalents, and then rearranging and modifying these new words according to the grammar of the target language.

It was soon found that this task was not easy, as a word can have several equivalents, and the correct one can be decided only by the context. The mathematical representation of the grammar of a natural language was also given up as an intractable problem. Machine translations were often incomprehensible, or they totally distorted meaning. By the mid-1960s the MT program as originally envisaged was dead.

Address correspondence to Subhash C. Kak, Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, Louisiana 70803.

International Journal of Approximate Reasoning 1987; 1:117–130

52 Vanderbilt Ave., New York, NY 10017 0888-613X/87/$3.50

The focus in recent years has shifted away from MT to obtaining preliminary translations that can then be worked upon by a human to produce the finished work. The other area to which attention has been given is the development of question/answer systems. These efforts have brought the problem of knowledge representation to the fore. Such a representation assumes knowledge of the application environment and of the intended audience. Various techniques for knowledge representation have been used in current systems, including semantic networks, first-order logic, frames, and production systems. Each of these techniques may offer special advantages for specific situations.

These techniques of knowledge representation also suffer from certain shortcomings. In the semantic network approach, where relations between objects and classes are specified, it is not known how to distinguish between information related to an object and to a class, nor is it known how to deal with exceptions. The first-order logic approach, which has led to the development of Prolog, cannot deal with incomplete knowledge or with situations where nondeductive inference may be called for. In the frame-based approach, the knowledge base is decomposed into highly modular chunks, although this procedure is not always possible. In the production system approach, rules connecting "patterns" and "action" (say, in human reasoning) are defined. Memory to define a pattern may be unavailable, however. The development of the so-called fifth-generation computers would be predicated on the success of the definition of "constrained" variants of natural languages and efficient knowledge representation systems.

If there are limits to the nature of natural language processing by the computer, the question arises: what is the nature of these limits? It appears that the best one can do to find these limits is to devise actual systems—in other words, use a constructive approach. The greatest success a constructive approach to the description of a natural language has ever had was when Dakshiputra Panini devised his grammar for the Sanskrit language, an achievement termed by the famous linguist Leonard Bloomfield [2] as "one of the greatest monuments of human intelligence." We would expect that the insights of Panini, "the greatest linguist of antiquity, if not of all time" (Staal [3]) could be exploited to help answer questions regarding limits to computers as well as to define an approach that could yield powerful text and speech processing systems. The knowledge representation methodology in the grammar of Panini and his successors is in many ways equivalent to the more powerful, currently researched artificial intelligence (AI) schemes. Furthermore, it includes rules about rules, analogs of which are not known for any other language (or for a flexible enough subset of, say, the English language), which would help in the writing of efficient AI software.

This article is a brief introduction to the Paninian approach, some aspects of which have already been incorporated in current computer understanding systems. This introduction also provides an overview of aspects of the

"standard" approach to natural language processing, so that readers may appreciate the commonality of the two as well as their main points of difference.

PANINI'S GRAMMAR

Dakshiputra Panini is believed to have lived during the fifth or sixth century B.C. He was born in the town of Shalatura, modern Lahur, in northwest India. Panini's grammar *Ashtadhyayi* (*The Eight Chapters*) deals ostensibly with the Sanskrit language; however, it presents the framework for a universal grammar that may (and probably does) apply to any language. His book consists of a little under 4000 rules and aphorisms. Panini's grammar attempts to completely describe Sanskrit as the spoken language of its time. Two important commentaries on his grammar that are often studied are those by Katyayana and Patanjali (second century B.C.). Its philosophical underpinnings were discussed in an important work in the fifth century A.D. by Bhartrhari. Modern translations and/or commentaries may be found in books by Bohtlingk [4], Renou [5], Kiparsky [6], Staal [3, 14], Scharfe [1, 15], and Misra [16].

Panini's grammar begins with meta-rules, or rules about rules. To facilitate his description he establishes a special technical language, or meta-language (Staal [14]). This is followed by several sections on how to generate words and sentences starting from roots, as well as rules on transformations of structure. The last part of the grammar is a one-directional string of rules, where a given rule in the sequence ignores all the rules that follow. Panini also uses recursion by allowing elements of earlier rules to recur in later rules. This anticipates in form and spirit by more than 2500 years the idea of a computer program. The structure of this part of Panini's grammar should rightly be termed the Panini Machine.

In Panini's system a finite set of rules is enough to generate an infinity of sentences. The algebraic character of Panini's rules was not appreciated in the West until recently when a similar generative structure was discussed by Noam Chomsky and others. Before this, in the nineteenth century, Panini's analysis of root and suffixes and his recognition of ablaut had led to the founding of the subjects of comparative and historical linguistics.

Despite similarities between Paninian and modern generative grammars, there exist striking differences as well. Some of these differences are related to the nature of the languages under study: Sanskrit and modern European languages. Furthermore, the contemporary evaluation of Panini is still going on and has been slowed by the fact that the original of his grammar is inaccessible to most linguists and computer scientists.

For the purpose of this article we would define an approach to language processing as being Paninian if it uses the following:

1. Root and affix analysis.
2. Linear strings of rules and analysis by rule sequence.

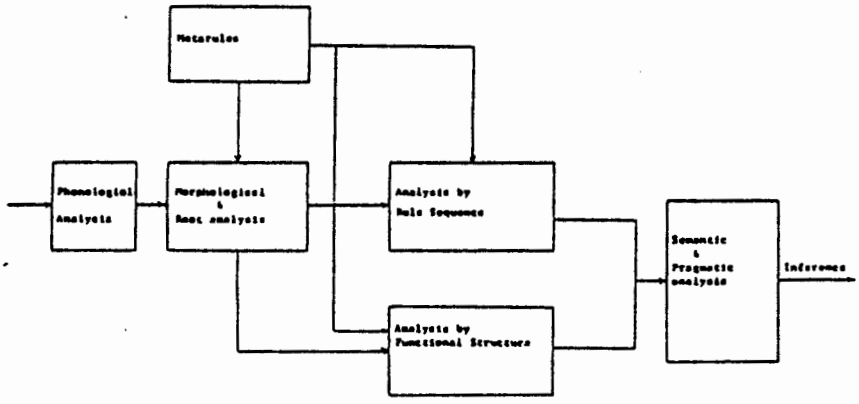

Figure 1. A Paninian Speech Understanding System

3. Analysis by functional structure.
4. An exhaustive description.

Because the current grammars for English do not satisfy condition (3), we would constrain the allowed structures so as to fit the capabilities of the grammars. It should be noted that as the significance of the Paninian structure for a universal grammar is further revealed and understood, our definition of this approach will change accordingly. Figure 1 is a block diagram of the basic elements of a Paninian speech understanding system.

LANGUAGE AMBIGUITIES

Several ambiguities make machine translation of languages a difficult task. These ambiguities need to be resolved, if at all possible, in various steps to obtain a translation. Each kind of ambiguity is addressed separately in a sequence of steps that constitutes the usual form of a computer-based understanding system.

Lexical Ambiguity

This type of ambiguity arises when a single word has two or more different meanings, all of which are potentially valid. Consider "Stay away from the range," which could be advice to keep away from either the stove or the meadow. "The court was packed" is a more complex example: "the court" may refer to a judicial or royal court, or a rectangular or open space, and "packed" might refer to a biased composition or a crowding by people.

Structural Ambiguity

One source of structural ambiguity is the many ways in which words in a sentence may be combined into phrases and then interpreted. Thus, in "He saw

the crane fly outside," one might be referring to the crane fly (a long-legged, two-winged fly) or to a bird known as a crane, flying. Other examples of ambiguity are: "My friend came home late last night" and "Flying kites can be tricky." Yet another kind of structural ambiguity occurs when the sentence has a unique grammatical structure but still allows different meanings because of different underlying "deep structures." For example, "The policeman's arrest was illegal" does not tell us who was arrested—the policeman or someone else. Another example is "That leopard was spotted."

Semantic Ambiguity

An example of this kind of ambiguity is the sentence "I like to eat brown grapes." This could either mean that the speaker liked a particular bunch of grapes in front of him, or that he merely expressed a preference for brown grapes.

Pragmatic Ambiguity

This ambiguity is related to the context of the sentence. Thus, in "She put the brick in the washer and spoiled it," the meaning would be different depending on whether the brick was made of metal or wax. Similarly, the meaning of "John loves his wife and so does Bill" is unambiguous only if it is known that Bill is a bachelor.

A further discussion of ambiguities of various kinds as well as approaches to theoretical and computational linguistics may be found in the references (see [7–13]). Such difficulties are inherent in English but are not fundamental to all natural languages. *Shastric* (scientific) Sanskrit is one natural language that appears to be particularly precise (Briggs [18]).

ANALYSIS OF LANGUAGE

A language may be described at several hierarchically organized levels. At the lowest level a spoken language may be characterized in terms of elementary sounds, the study of which constitutes *phonology*. A group of similar speech sounds that function in the same way, and that may be substituted for each other without changing the meaning of an utterance, is called a *phoneme*. Natural languages are generally characterized by 30 to 80 phonemes. English is usually described in terms of about 40 phonemes. Sanskrit is described in terms of 48 phonemes, each of which is represented by a unique symbol in its alphabet (13 vowels and 35 consonants). Each basic unit of the written language, which includes letters of the alphabet, the punctuation marks, and the blanks separating words, is called a *grapheme*.

The study of the next level of linguistic analysis is termed *morphology*. Simple words as well as inflectional sounds such as plural endings or prefixes and suffixes that convey meanings are *morphemes*. Written words must be analyzed for their morphemic components in a natural language processing system. Thus, a system might process "unknowing" by finding its root form "know" and then determining the change in meaning effected by each of the additional morphemes "un-" and "-ing."

Syntax deals with the manner in which the meaningful constituents are put together to form an utterance or sentence. The structure is usually represented in terms of a tree. The sentence "I like to eat the fruit from my garden" may be represented by a parse tree, as in Figure 2.

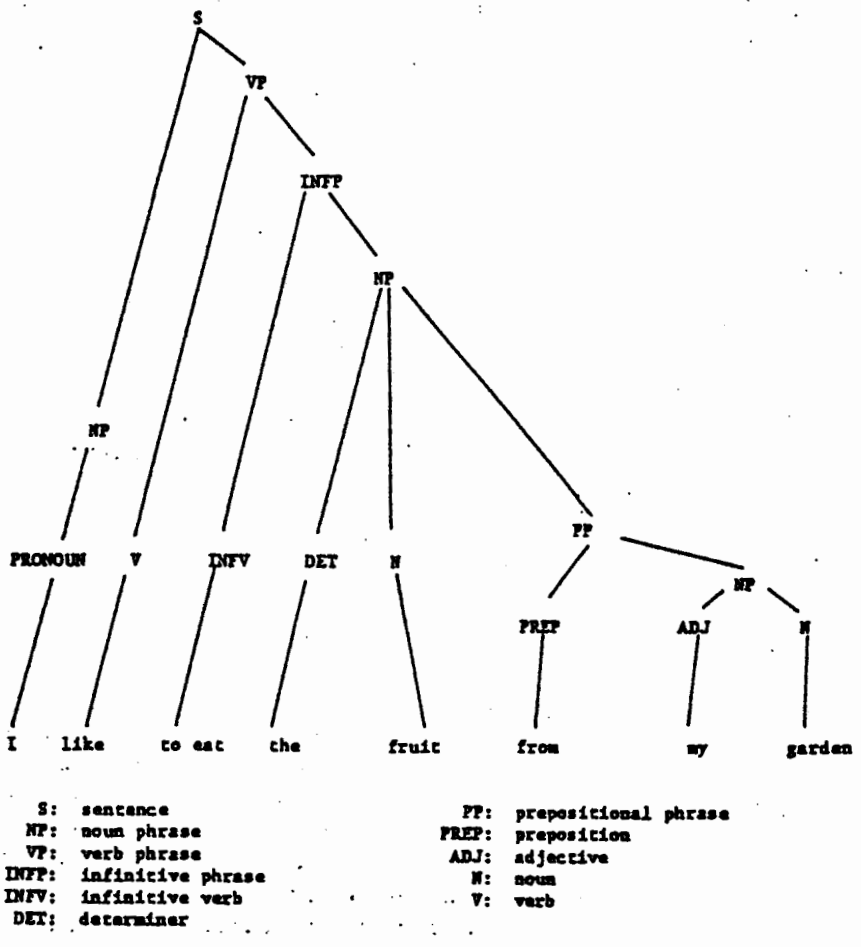

Figure 2. A Parse Tree

Some sentences can be parsed in many different ways, leading to different meanings. As an example, "Traffic jams were caused by slow trucks and buses carrying heavy loads" may be parsed in at least four different ways. In these different parse trees, "slow" may qualify trucks only or trucks and buses; similarly, "heavy loads" may qualify buses only, or both trucks and buses.

A parse tree is generated by rules of the kind

$$S \rightarrow NP + VP$$

$$NP \rightarrow ADJ + N$$

$$NP \rightarrow DET + N + PP$$

$$PP \rightarrow PREP + NP$$

and so on. A collection of such rules is one way to generate a grammar. Most traditional English grammars have this form.

The grammatical tradition that grew out of Panini's work did not analyze sentences in the above noun-phrase/verb-phrase form. Rather, the description was a generative one; the structure of the sentence was derived from a number of primitive syntactic categories such as verbal action, agents, objects, and so on.

An early concern of grammarians was how a serial act, such as a string of words in a sentence, can communicate a unitary sense, the meaning. The Sankhya system, the dominant philosophical school of the day, was based on a principle of enumeration that allowed consideration of time to be discrete and not infinitely divisible, as it was viewed by the Greeks. Having taken this premise, there was no question of Zeno's paradoxes of time and motion arising in this framework. Also, a serial representation of meaning echoed the categorization of reality as in Sankhya.

Early grammatical work focused on rules of sandhi. This appears to have been motivated by a desire to keep the utterance of the Vedas constant. In this work, three kinds of rules were postulated: universal rules, exceptional rules, and counter rules. Meta-rules were also defined. One meta-rule posited that the exceptional rule is stronger than the universal where both are applicable. The counter rule freed the subject from the domain of the universal rule construct. This structure incorporated profound insights into the nature of generative systems, especially with regard to defining exceptions and considering rules about rules.

The meaning of an utterance was given a central place in an early grammar usually attributed to Indra. He is also said to have expressed the property of words having a structure of root and termination. Bharadvaja appears to have been the first to declare that the verb is central to meaning. This is not surprising, as verb denotes action or change, which is the cornerstone of Sankhya. Yaska said that the verb and the noun have becoming and being as their fundamental notions.

Panini took the idea of action as defined by the verb and developed a comprehensive theory by providing a context for action in terms of its relations to agents and situation. This theory is called the *karaka* theory. Panini introduced six basic semantic notions that capture several aspects of action through its participants. These karakas are as follows (the numbers in parentheses refer to the rule numbers):

Apadana:	that which is fixed when departure takes place (1.4.24)
Sampradana:	the recipient of the object (1.4.32)
Karana:	the main cause of the effect; instrument (1.4.42)
Adhikarana:	the basis, location (1.4.45)
Karman:	what the agent seeks most to attain; deed, object (1.4.49)
Kartr:	one who is independent; the agent (1.4.54)

These karakas do not always correspond with the nature of an action; therefore, the karaka theory is only a *via media* between grammar and reality. It is general enough, however, to subsume a large number of cases, and where not directly valid, the essence of the action/transaction can still be cast in the karaka mold. To do the latter, Panini requires that the intent of the speaker be considered. Rather than a structure based on conventions regarding how to string together words, Panini's system is based on meaning. It should also be noted that the karakas do not have a one-to-one correspondence with grammatical cases.

COMPUTER UNDERSTANDING OF TEXT

A computer-based system for speech or text understanding must perform several operations in sequence. For speech the first operation is that of phonological analysis, in which sound waves are transcribed into a series of phonemes. These may then be expressed as written words. The first analysis performed for written language is morphological, where each word is decomposed into its root and inflections. This is followed by lexical analysis, where the words are assigned to different lexical categories such as noun, verb, adjective, and so forth. The next operation is that of syntactic analysis or parsing, where the rules of grammar are used to yield the structure of the sentence. The next step is that of semantic analysis, where the sentence is converted into a form such that inferences can be drawn easily. The last step is that of pragmatic analysis, which makes the context of the sentence explicit. At the end of the sequence, the computer can announce its inferences and respond to questions.

For such a sequence of operations to proceed effectively, it is necessary that the meaning underlying the sentences be represented in a convenient form. Different knowledge representation systems have been devised for this purpose. The sequence of operations in the understanding system is then a means of first

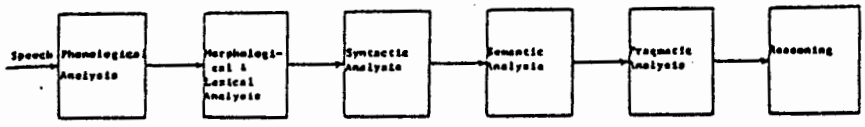

Figure 3. A Basic Speech Understanding System

generating information about the knowledge representation system, which is followed by analysis.

Figure 3 is a representation of a typical computer understanding system. Note that the Paninian system of Figure 1 captures the essentials of such a system while further specifying some operations.

Until now the most effective parsers have been the ones that are based on grammars designed with computers in mind. An early system (developed in 1971) that was quite impressive in realizing its rather limited aims was Terry Winograd's SHRDLU, in which the parsing was done by interpreting grammars written as programs. SHRDLU simulates a robot that lives in a tabletop world containing a box and several colored pyramids, blocks, and cubes of assorted sizes. SHRDLU keeps track of the locations of the blocks and can pick them up and rearrange them. The system accepts natural English input and responds by executing commands and answering queries about this world of blocks.

Syntactic classification in SHRDLU is done in terms of unanalyzable markings (features). The interdependence of these features can be represented by a graph. Each lexical entry for a word consists of a list of features and a semantic entry, the latter being a data structure that acts as the "meaning" of the word. The parser, which is written in a language called PROGRAMMAR, builds a syntactic tree where each node has a category label (such as noun, verb phrase, etc.), a list of features, and an associated semantic structure. SHRDLU was a successful system because its domain was very limited: a few actions with a set of blocks.

A representation of SHRDLU is given in Figure 4. MONITOR initiates and terminates the processing of an input sentence. INPUT, together with DICTIONARY, carries out a morphemic analysis of the input sentences and provides GRAMMAR with strings of words. PROGRAMMAR does the parsing of sentences, and ANSWER converts the system response into grammatical English and keeps track of context. PLANNER is used for analysis and deduces facts about the environment. BLOCKS is a subset of PLANNER theorems that embody the system's knowledge about its physical environment, and DATA contains PLANNER statements that describe the object in the scene being scanned. This system is thus a special implementation of the general system of Figure 3.

Other systems that deal with domains much larger than or different from that

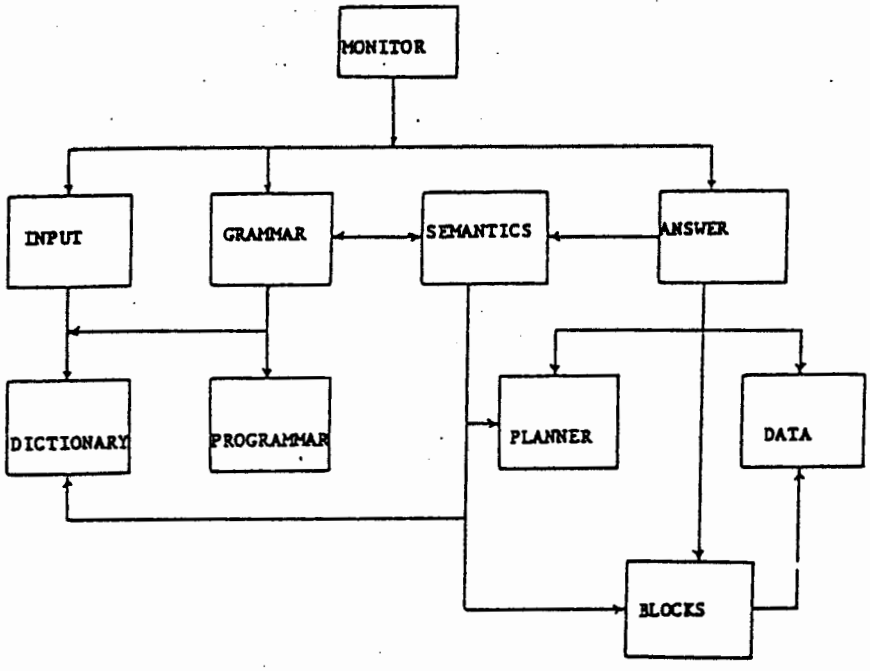

Figure 4. Winograd's SHRDLU

of SHRDLU have been designed in recent years. Many of these constitute parts of expert systems. They also use a variety of grammars.

REPRESENTING GRAMMARS

A grammar may be coded in terms of a network where transitions between the constituents of a sentence are coded as transitions across the nodes. The arcs in the network contain additional information that instruct the parser as to what action it should take in order to generate a specific meaning structure. Such networks are called augmented transition networks (ATNs).

The ATN grammar is a standardized set of tests and operations that are performed on each sentence. This can be represented pictorially by a directed graph, where arcs represent test-operation pairs and nodes indicate common points joining arcs. The tests are generally conditions on an input word, and the network represents the several ways a sentence can be analyzed. All operations are carried out in order, and a complete analysis is produced by a path through the network on which all the tests are satisfied by the input words. Another useful grammar is the lexical function grammar (LFG). In LFG, grammatical

functions are expressed in a form so that categories such as head, number, person, tense, subject, object, and so on are tied to the words and phrases that serve these functions. Such a categorization is a part of the Sanskrit grammar of Panini, where modifiers identify subject and object, for example, and word order is free (Staal [14]) and dictated mainly by style and convention. The LFG approach allows a sentence to be represented in a nested form if a part of it plays a role in another part.

For semantic analysis one needs to represent the sentence in a form so that reasoning procedures can be applied to it. One may use the semantic network approach to classify objects through their relationships, which information can then be used with predicate calculus for further processing for the computer to draw its inferences. An example of a simple semantic network is given in Figure 5. Such a network is often at the basis of most knowledge representation systems. Another semantic network that expresses knowledge in a form literally

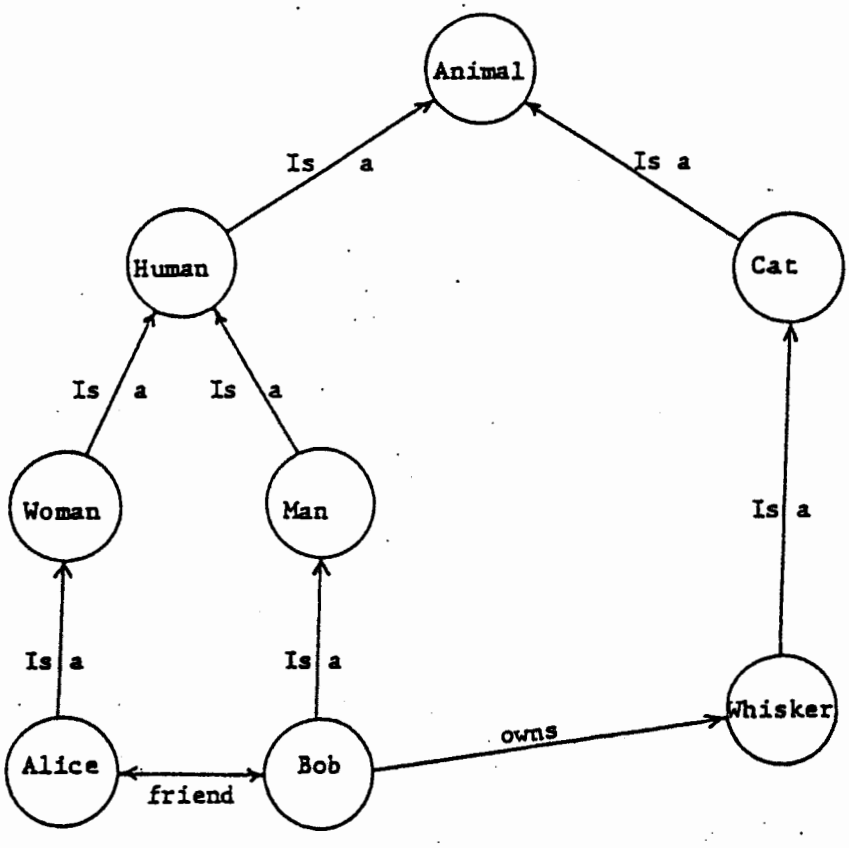

Figure 5. A Simple Semantic Network

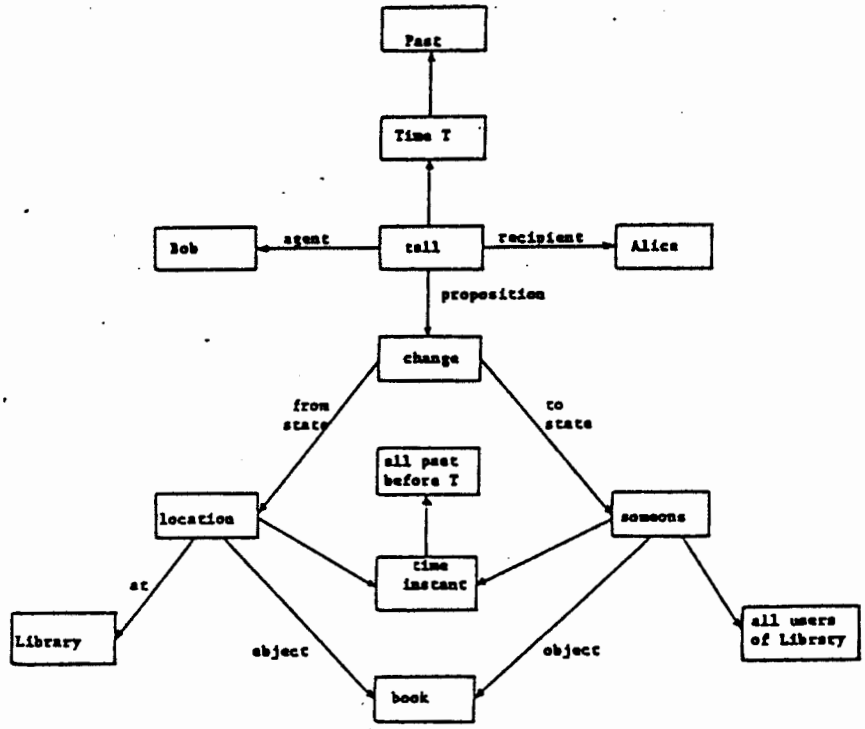

Figure 6. Bob Told Alice that the Book Had Been Checked Out of the Library

identical to that of Panini and other Sanskrit grammarians (Briggs [18]) is given in Figure 6. This example represents the web of changes through agents and across space and time for the given object.

The analysis in the grammars of Panini and his successors is done using the karaka theory by analyzing the forms of the various words in the sentence. The transformations define agents and recipients as well as time and space relationships.

For example, consider "Ratrau tvya mathmadhye na praveshtavyam," which means "At night you must not enter the monastery." The future passive participle is formed with the suffix *tvya,* which transforms *pravish* (enter). *Ratri* (night) is modified to express an event at night, and *mathmadhye* (within the monastery) represents the dative singular; *tvya* is you in the instrumental form, and *na* is not. This shows how the action, agent, and time aspects are made clear.

Briggs has discussed more fully the question of equivalence between knowledge representation in AI models and in Sanskrit grammar.

SUMMARY

The current natural language understanding systems have evolved into a form that has certain attributes of the Paninian approach. Current systems appear to be promising only in severely restricted universes of objects and relationships. It appears that to make significant progress, a two-pronged attack on the problem is required: (1) restricting the domain of allowed grammatical relationships in the language and (2) expanding the understanding of the grammar by drawing upon the insights of Panini, as summed up in his grammar.

A comparison of the work of later Indian grammarians such as Nagesha Bhatta (1730–1810) and various recent knowledge representation schemes of AI has been given by Briggs [18], who stated: "Among the accomplishments of the grammarians can be reckoned a method for paraphrasing Sanskrit in a manner that is identical not only in essence but in form with current work in Artificial Intelligence. [And] a natural language [can] serve as an artificial language also, and that much work in AI has been reinventing a wheel millenia old."

To summarize, the current knowledge representation systems of AI agree with the requirements of the Paninian approach. This makes analysis systematic once the knowledge in a text has been represented. However, this does not answer the question of a successful extraction of knowledge because, as has been indicated, natural language is full of various kinds of ambiguity. There are two different ways one can face this issue squarely. One may use Sanskrit as an intermediary natural language because its grammar is exhaustive. This is unlikely to happen owing to the difficulty of learning this language. The other way is to seek new, generative, Paninian-style grammars for the English language.

It has been argued by Chomsky and others that the process of language acquisition by children suggests that the generative principles of universal grammar must be innate to the human mind, and that "the general features of language structure reflect not so much the course of one's experience, but rather the general character of one's capacity to acquire knowledge—in the traditional sense, one's innate ideas and innate principles" (Chomsky [19]). This argument implies that it should be possible to find Paninian grammars for English and other modern languages. As is well known, Panini was driven by considerations of finding the smallest set of rules and meta-rules (Shefts [17], Kak [20]). A development of a Paninian-style grammar would then involve a search for generative rules and meta-rules.

References

1. Scharfe, H., *Grammatical Structure*, Otto Harrassowitz, Wiesbaden, Germany, 1977.

2. Bloomfield, L., *Language*, Holt, Rinehart, New York, 1933.

3. Staal, J. F. (Ed.). *A Reader on the Sanskrit Grammarians*, MIT Press, Cambridge, Mass., 1972.

4. Bohtlingk, O., *Panini's Grammatik*, George Olms, Hildesheim, Germany, 1964.

5. Renou, L., *La Grammaire de Panini*, C. Klincksieck, Paris, 1966.

6. Kiparsky, P., *Panini as a Variationist*, MIT Press, Cambridge, Mass., 1979.

7. Lyons, J., *Introduction to Theoretical Linguistics*, Cambridge U.P., England, 1968.

8. Lyons, J. (Ed.). *New Horizons in Linguistics*, Penguin, New York, 1970.

9. Schank, R. C. and Colby, K. M. (Eds.). *Computer Models of Thought and Language*, Freeman, San Francisco, 1973.

10. Winograd, T., *Language as a Cognitive Process*, Addison-Wesley, Reading, Mass., 1983.

11. Tennant, H., *Natural Language Processing*, Petrocelli, Princeton, N.J., 1981.

12. Bresnan, J. (Ed.). *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Mass., 1982.

13. Ritchie, G. D., *Computational Grammar*, Barnes & Noble, Totowa, N.J., 1980.

14. Staal, J. F., *Word Order in Sanskrit and Universal Grammar*, D. Reidel, Dordrecht, Holland, 1967.

15. Scharfe, H., *Panini's Metalanguage*, American Philosophical Society, Philadelphia, 1971.

16. Misra, V. N., *The Descriptive Technique of Panini*, Mouton, The Hague, The Netherlands, 1966.

17. Shefts, B., *Grammatical Method in Panini*, American Oriental Society, New Haven, Conn., 1961.

18. Briggs, R., "Knowledge representation in Sanskrit and artificial intelligence," *AI Magazine* 6, 22–38, 1985.

19. Chomsky, N., *Aspects of the Theory of Syntax*, MIT Press, Cambridge, Mass., 1965.

20. Kak, S., *The Nature of Physical Reality*, Peter Lang, New York, 1986.